Importación de Datasets

Sesión 01

Ing. Gómez Marín, Jaime[1]

Módulo 2 : Estadística y Visualización de Datos con Python
Departamento de TdG

August 2019

Índice

Introducción

En esta sesión se aprenderá a usar la librería Numpy, el cuál agrega características avanzadas de Python para el manejo de arreglos y matrices. Numpy facilita una gran repertorio de funciones matemáticas de alto nivel.

Librería Numpy

Es una librería de Python escrita para la manipulación de arreglo y matrices, tambien tiene un gran repertorio de funciones matemáticas avanzadas :

- Operaciones básicas de arreglos y matrices
- Indexaciones, Slicing e interacciones en matrices,
- Condiciones y operaciones booleanas
- Gestiones de arreglos.

Objeto Ndarray

El elemento principal de la librería Numpy es el objeto Ndarray. Este objeto es un arreglo multidimensional homogéneo con un predeterminado número de elementos.
La manera más fácil de definir un nuevo ndarray es usando una función array(), en la cuál se pasa una lista con todos los elementos que formaran el arreglo Ndarray.

Code : Ndarray

```
1  '''
2  Programa : Ciencia de Datos con Pthon
3  Modulo 02 : Estadística y Visualización de Datos con
4  Sesion 01 : Importación de Datasets
5  Fecha : 18/08/2019
6  Version : 1
7  Author : Jaime Gomez
8  '''
9
10 import numpy as np
11
12 print("-----------------------------------")
13
14 # Define listado
15 nro_list = [1,2,3,4,5,6]
16 nro_array = np.array(nro_list)
```

Code : Ndarray

```
1 print(type(nro_array))
2 print("nro_array --> ",nro_array)
3 print("dtype --> ",nro_array.dtype)
4 print("ndim --> ",nro_array.ndim)
5 print("size --> ",nro_array.size)
6 print("shape --> ",nro_array.shape)
7
8 print("itemsize --> ",nro_array.itemsize)
9 print("data --> ",nro_array.data)
```

Code : Ndarray: Creación de arrays

```
1  # Create arrays
2  print("-----------------------------------")
3
4  nro_array = np.array([[1,2,3],[4,5,6]])
5  print(nro_array)
6
7  nro_array = np.array(((1,2,3),(4,5,6)))
8  print(nro_array)
9
10 nro_array = np.array([(1,2,3),(4,5,6)])
11 print(nro_array)
```

Code : Ndarray: Otras formas de creación de arrays

```
1  ceros_array = np.zeros((3,3))
2  print("np.zeros((3,3)) --> ",ceros_array)
3
4  unos_array = np.ones((3,3))
5  print("np.ones((3,3)) --> ",ceros_array)
6
7  nros_array = np.arange(8)
8  print("np.arange(8) --> ",nros_array)
9
10 nros_array = np.arange(3,8)
11 print("np.arange(3,8) --> ",nros_array)
12
13 nros_array = np.arange(1,16,3)
14 print("np.arange(1,16,3) --> ",nros_array)
```

Code : Operaciones Básicas - 1

```
1  '''
2  Programa : Ciencia de Datos con Pthon
3  Modulo 02 : Estadística y Visualización de Datos con
4  Sesion 01 : Importación de Datasets
5  Fecha : 18/08/2019
6  Version : 1
7  Author : Jaime Gomez
8  '''
9
10 import numpy as np
11
12 # Basic Operations
13 print("-----------------------------------")
14
15 # Arithmetic Operators
16 a_array = np.arange(4)
17 print(a_array)
```

Code : Operaciones Básicas - 2

```
1  # Arithmetic Operators
2  a_array = np.arange(4)
3  print(a_array)
4  print(a_array + 4)
5  print(a_array * 4)
6
7  b_array = np.arange(4,8)
8  print(b_array)
9  print(b_array + 4)
10 print(b_array * 4)
```

Code : Operaciones Básicas - 3

```
1  print("-------------------------------------")
2  print(a_array)
3  print(b_array)
4  print("a_array + b_array \n",a_array + b_array)
5  print("a_array - b_array \n",a_array - b_array)
6  print("a_array * b_array \n",a_array * b_array)
7
8  print("-------------------------------------")
9  print("a_array : \n",a_array)
10 print("a_array*5 : \n",a_array * 5)
```

Code : Operaciones Básicas - 4

```
1  # The Matriz Product
2  print("-------------------------------------")
3  c_matriz = np.dot(a_matriz,b_matriz)
4  print("a_matriz : \n",a_matriz)
5  print("b_matriz : \n",b_matriz)
6  print("np.dot(a_matriz,b_matriz) : \n", c_matriz )
7
8  print("-------------------------------------")
9  c_matriz = a_matriz.dot(b_matriz)
10 print("a_matriz : \n",a_matriz)
11 print("b_matriz : \n",b_matriz)
12 print("a_matriz.dot(b_matriz) : \n", c_matriz)
```

Code : Operaciones Básicas - 5

```
1  # Universal Functions (ufunc)
2  print("------------------------------------")
3
4  a_array = np.arange(4,8)
5  print("a_array : \n",a_array)
6  print("np.pow(a_array,2) : \n",np.power(a_array,2))
7
8  # Aggregate Functions
9
10 print("------------------------------------")
11
12 a_array = np.arange(4,8)
13 print("a_array : \n",a_array)
14 print("a_array.sum() : ",a_array.sum())
15 print("a_array.min() : ",a_array.min())
16 print("a_array.max() : ",a_array.max())
17 print("a_array.mean(): ",a_array.mean())
18 print("a_array.std() : ",a_array.std())
```

Code : Indexing

```
1     print("value --> ",value)
2
3  print("-----------------------------------")
4  a_matriz = np.arange(0,12).reshape(3,4)
5  print("a_matriz : \n",a_matriz)
6
7  # Procesa el promedio de la matriz por columna
8  a_m = np.apply_along_axis(np.mean,axis=0
9                            ,arr=a_matriz)
10 print(a_m)
11
12 # Procesa el promedio de la matriz por fila
13 a_m = np.apply_along_axis(np.mean,axis=1,
```

Code : Slicing -1

```
print("-----------------------------------")
a_array = np.arange(0,10)
print("a_array          : ",a_array)
print("a_array[0:5]     : ",a_array[0:5])
print("a_array[0:5:2]   : ",a_array[0:5:2])
print("a_array[::2]     : ",a_array[::2])
print("a_array[:5:2]    : ",a_array[:5:2])
print("a_array[:5:]     : ",a_array[:5:])
print("a_array[::3]     : ",a_array[::3])
```

Code : Slicing - 2

```
1
2  print("-----------------------------------")
3  a_matriz = np.arange(0,12).reshape(3,4)
4  print("a_matriz : \n",a_matriz)
5  print("a_matriz[:,0] : \n",a_matriz[:,0])
6  print("a_matriz[0,:] : \n",a_matriz[0,:])
7  print("a_matriz[0:2,0:2] : \n",
8        a_matriz[0:2,0:2])
9  print("a_matriz[[0,2],0:2] : \n",
10       a_matriz[[0,2],0:2])
```

Code : Iterating - 1

```
1  print("-----------------------------------")
2  a_matriz = np.arange(0,12).reshape(3,4)
3  print("a_matriz : \n",a_matriz)
4
5  # Procesa el promedio de la matriz por columna
6  a_m = np.apply_along_axis(np.mean,axis=0
7                            ,arr=a_matriz)
8  print(a_m)
9
10 # Procesa el promedio de la matriz por fila
11 a_m = np.apply_along_axis(np.mean,axis=1,
12                           arr=a_matriz)
13 print(a_m)
```

Code : Iterating - 2

```
1  print("-----------------------------------")
2  a_matriz = np.arange(0,12).reshape(3,4)
3  print("a_matriz : \n",a_matriz)
4
5  # Definamos funcion al cuadrado
6  def potencia(x):
7      return x*x
8
9  # Procesa el promedio de la matriz por columna
10 a_m = np.apply_along_axis(potencia, axis=0,
11                           arr=a_matriz)
12 print(a_m)
13
14 # Procesa el promedio de la matriz por fila
15 a_m = np.apply_along_axis(potencia, axis=1,
16                           arr=a_matriz)
17 print(a_m)
```

Code : Condicionales y expresiones booleanas

```
1  print("-----------------------------------")
2  a_matriz = np.random.random((4,4))
3  print(a_matriz)
4
5  #print(np.round(1.335,decimals=2))
6
7  a_m = np.apply_along_axis(np.round,
8                  axis=1, arr=a_matriz, decimals=2)
9  print(a_m)
10
11 # Se muestran la tabla booleana
12 print( a_m < 0.5)
13
14 # Solo se muestran los que cumplen la condicion
15 print( a_m[a_m < 0.5])
```

Code : Unificar matrices

```
1  print("-----------------------------------")
2  a = np.ones((3,3))
3  print(a)
4  print("     ")
5
6  b = np.zeros((3,3))
7  print(b)
8  print("     ")
9
10 # Incrementa la cantidad de filas
11 c = np.vstack((a,b))
12 print(c)
13 print("     ")
14
15 # Incrementa la cantidad de columnas
16 d = np.hstack((a,b))
17 print(d)
```

Code : Unificar arreglos

```
print("-----------------------------------")
dat_1 = np.array([1,2,3])
dat_2 = np.array([4,5,6])
dat_3 = np.array([7,8,9])
print(dat_1)
print(dat_2)
print(dat_3)

# Unificar arreglos unidimensionales en columnas
col_datos \
    = np.column_stack((dat_1,dat_2,dat_3))
print(col_datos)
print("     ")

# Unificar arreglos unidimensionales en filas
filas_datos \
    = np.row_stack((dat_1,dat_2,dat_3))
print(filas_datos)
```

Code : Referencia a objetos

```
1  # References of Objects
2  print("-------------------------------------")
3  a = np.array([1, 2, 3, 4])
4  # Se asigna una referencia
5  b = a
6
7  print("Antes de hacer cambios")
8  print("a :",a)
9  print("b :",b)
10
11 a[2] = 12
12 print("Despues de hacer cambios")
13 print("a :",a)
14 print("b :",b)
```

Code : Referencia a arreglos

```
1  # Views of Objects
2  print("-----------------------------------")
3  a = np.array([1, 2, 3, 4])
4  # Se crea una vista
5  c = a[1:3]
6
7  print("Antes de hacer cambios")
8  print("a :",a)
9  print("c :",c)
10
11 a[1] = 11
12
13 print("Despues de hacer cambios")
14 print("a :",a)
15 print("c :",c)
```

Code : Copia de objetos

```
1  # Copies of Objects
2  print("-----------------------------------")
3  a = np.array([1, 2, 3, 4])
4  # Se realiza una copia
5  b = a.copy()
6  print("Antes de hacer cambios")
7  print("a :",a)
8  print("b :",b)
9
10 a[2] = 12
11 print("Despues de hacer cambios")
12 print("a :",a)
13 print("b :",b)
```

Code : Broadcasting

```
1 # Broadcasting
2 print("-----------------------------------")
3 A = np.arange(16).reshape(4, 4)
4 b = np.arange(4)
5 C = A + b
6
7 print("A    : \n",A)
8 print("b    : \n",b)
9 print("A+b  : \n",C)
```

Code : Estructura - 1

```
1  # Structured Arrays
2
3  print("-----------------------------------")
4  estructura \
5      = np.array([(1,"Uno",0.5),
6                  (2,"Dos",1.5),
7                  (3,"Tres",2.5)])
8
9  print(estructura.dtype)
10 print(estructura)
11
12 '''
13 bytes                    b1
14 int                      i1, i2, i4, i8
15 unsigned ints            u1, u2, u4, u8
16 floats                   f2, f4, f8
17 complex                  c8, c16
18 fixed length strings     a<n>
19 '''
```

Code : Estructura - 2

```
1  print("-----------------------------------")
2  estructura \
3      = np.array([(1,"Uno",0.5),
4                  (2,"Dos",1.5),
5                  (3,"Tres",2.5)],
6                 dtype=('i2,a6,f4'))
7
8  print(estructura.dtype)
9  print(estructura)
10
11 print("estructura[0] : "
12       ,estructura[0])
13 print("estructura['f0'] : "
14       ,estructura['f0'])
```

Code : Estructura - 3

```
1  print("-----------------------------------")
2  estructura \
3      = np.array([(1,"Uno",0.5),
4                  (2,"Dos",1.5),
5                  (3,"Tres",2.5)],
6             dtype=[('id','i2'),
7                    ('orden','a6'),
8                    ('valor','f4')])
9
10 print(estructura.dtype)
11 print(estructura.dtype.names)
12 print(estructura)
13
14 print("estructura[0] : ",
15                 estructura[0])
16 print("estructura['orden'] : ",
17                 estructura['orden'])
```

Code : Leer y Escribir archivos binarios

```
1  import numpy as np
2
3  # Loading and Saving Data in Binary Files
4  print("-----------------------------------")
5  NOMBRE_ARCHIVO = 'archivo_binario.npy'
6  datos = np.random.random(9).reshape((3,3))
7  print("datos : \n",datos)
8
9  np.save(NOMBRE_ARCHIVO,datos)
10 datos_cargados = np.load(NOMBRE_ARCHIVO)
11
12 print("datos_cargados : \n",datos_cargados)
```

Code : Leer archivos CSV

```
1  # Reading files CSV
2  print("-----------------------------------")
3  NOMBRE_ARCHIVO = \
4      'medallero_Panamericanos_Lima2019.csv'
5  datos = \
6      np.genfromtxt(NOMBRE_ARCHIVO, delimiter=','
7                    ,names=True, dtype=None)
8  print(datos)
9
10 print("-----------------------------------")
11 NOMBRE_ARCHIVO = \
12     'medallero_Panamericanos_Lima2019.csv'
13 datos = \
14     np.genfromtxt(NOMBRE_ARCHIVO, delimiter=','
15                   ,names=True, dtype=None
16                   ,encoding="utf-8")
17 print(datos)
```

Conclusion

En esta sesión se han tocado temas acerca de los principales aspectos de la librería NumPy que serán usados en las siguientes sesiones de este programa. Pandas es una librería que se basa en NumPy

Bibliography

 Fabio Nelli. Python Data Analytics with Pandas, NumPy, and Matplotlib, 2018.